Randomized Algorithms for Dimension Reduction on Massive Data

Stoyan Georgiev and Sayan Mukherjee[*]

November 8, 2012

Abstract

A variety of dimension reduction methods can be framed as (generalized) eigenvalue decomposition problems. For these methods we develop an algorithmic framework for inference of low-dimensional structure from massive high-dimensional data by leveraging recently developed highly scalable randomized low-rank approximation approaches. We propose efficient randomized algorithms for supervised and unsupervised (non)linear dimension reduction - specifically, (localized) Sliced Inverse Regression (SIR), Locality Preserving Projections (LPP), and Principal Components Analysis (PCA). A key point in our development is the emphasis on the dual role of randomization as simultaneously enhancing the computational feasibility while introducing regularization in the dimension reduction estimators. This point is highlighted by results on simulated and real data that show improved performance for the randomized over the exact methods.

Key Words: dimension reduction, randomized algorithms, random projections, Krylov subspace methods, supervised dimension reduction

[*]Stoyan Georgiev is a Post-doctoral Scholar in Human Genetics at the University of Chicago, Chicago, IL 60637, U.S.A. (Email:sgeorg80@gmail.com) Sayan Mukherjee is an Associate Professor in the Departments of Statistical Science, Computer Science and Mathematics, and the Institute for Genome Sciences & Policy, Duke University, Durham, NC 27708-0251, U.S.A. (Email:sayan@stat.duke.edu).

1 Introduction

In the current era of information, large amounts of complex high dimensional data are routinely generated across science and engineering. Understanding the underlying structure in the data and using this structure to model scientific problems is of fundamental importance in a variety of applications. As the size of data sets increases the problem of statistical inference and computational feasibility become inextricably linked. The idea of dimension reduction is a natural approach to summarize massive data and has historically played a central role in data visualization, predictive modeling and general data analysis in both statistical inference (Adcock, 1878; Edegworth, 1884; Fisher, 1922; Hotelling, 1933; Young, 1941) and numerical analysis (Golub, 1969; Golub and Van Loan, 1996; Gu and Eisenstat, 1996; Golub et al., 2000), for a recent review see Mahoney (2011). Historically, statisticians have focused on procedures and theory for estimators, for example the subspace that captures the maximal variation in data drawn from a population. Numerical analysis, or computational mathematics, is then used to provide efficient algorithms, with provable stability and convergence guarantees, to compute these estimates. A classic example of this interplay is Principal Components Analysis (PCA) (Hotelling, 1933). In PCA a target estimator is defined based on statistical considerations about the sample variance in the data, which is then efficiently computed using a variety of Singular Value Decomposition (SVD) algorithms developed by the numerical analysis community. In this paper we focus on the problem of dimension reduction integrating the statistical considerations of *estimation accuracy* and out-of-sample errors with the numerical considerations of runtime and *numerical accuracy*. Given a very large high-dimensional data set a natural approach is to try and compress it with minimal loss of information prior to statistical analysis. In this paper we focus on dimension reduction as a means of compression and out-of-sample residual errors as a measure of information. This idea underlies a variety of approaches to linear and non-linear dimension reduction both in the unsupervised and the supervised setting. When the goal is prediction of a quantitative response, dimension reduction is often coupled with regression. Many dimension reduction methods (Hotelling, 1933; Fisher, 1936; Li, 1991; Wu et al., 2010; Belkin and Niyogi, 2003; Donoho and Grimes, 2003; Roweis and Saul, 2000) can be reduced to the solution of generalized eigendecomposition problems. These can be efficiently solved when either the number of observations, n, or the number of dimensions, p, is small. However, when both dimensions are large the typical computational cost of $O(n^2p)$ of current algorithms that rely on eigendecomposition of the data matrix becomes prohibitive. The numerical analysis community has recently developed powerful randomized algorithms for approximate solutions to eigendecomposition problems (Drineas et al., 2006; Sarlos, 2006; Liberty et al., 2007; Boutsidis et al., 2009; Rokhlin et al., 2008; Halko et al., 2009), including applications to efficient implementation of PCA. Analysis of these randomized algorithms has focused on the difference between the approximate and the exact decomposition of the observed data matrix. This misses the very important point that the observed data is a random realization from a population and our interest is in a comparison of the approximate decomposition of the observed data to that of the population. From this perspective the approximate decomposition not only plays an important computational role but also has an impact on the inference properties of the resulting estimators. One objective of this paper is to demonstrate that randomization serves as a regularizer and can help avoid over-fitting to the noise in the observed data. In this paper we will develop novel randomized algorithms for dimension reduction that scale to massive data and discuss the relation between randomization and the classic notions of regularization or shrinkage. We will focus on key parameters in these numerical algorithms which have both runtime implications as well as a natural interpretation as regularizers of the final estimators. Specifically, we develop randomized algorithms for a variety of dimension reduction methods including: PCA as a linear unsupervised method, Sliced Inverse Regression (SIR) (Li, 1991) as a linear supervised method, Localized Sliced Inverse Regression (LSIR) (Wu et al., 2010) as a supervised non-linear dimension reduction method, and Locality Preserving Projections (LPP) (He and Niyogi, 2003) a non-linear unsupervised method. The framework used to develop these randomized algorithms is general and can be extended to a variety of dimension reduction methods (Tenenbaum et al., 2000; Roweis and Saul, 2000; Donoho and Grimes, 2003; Belkin and Niyogi, 2003; Li, 1991; Cook and Weisberg, 1991; Li, 1992; Hastie and Tibshirani, 1996; Goldberger et al., 2005; Globerson and Roweis, 2006; Li et al., 2005; Nilsson et al., 2007; Sugiyama, 2007; Cook, 2007; Wu et al., 2010).

2 Randomized algorithms for dimension reduction

We will develop randomized algorithms for four dimension reduction methods: PCA, SIR, LSIR, and LPP. The first algorithm will provide a numerically efficient and statistically robust estimate of the principal components from data based on a randomized algorithm for singular value decomposition (RSVD) (Rokhlin et al., 2008; Halko et al., 2009) . In this problem the objective is linear unsupervised dimension reduction with the low-dimensional subspace estimated via an eigendecomposition. RSVD will serve as the core computational engine for the other three dimension reduction algorithms in which estimation reduces to a generalized eigendecomposition problem. The second algorithm computes a low dimensional linear subspace that captures the predictive information in the data. This is a supervised setting, in which the input data consist of a set of features and a response variable for each observation. We decided to focus on SIR (Li, 1991) as it applies to both categorical and continuous responses and subsumes the widely used Linear Discriminants Analysis (LDA) (Fisher, 1936) as a special case. The third method to which we apply randomization ideas is LSIR (Wu et al., 2010), which is an extension of SIR to the non-linear supervised dimension reduction setting. Finally, we develop a randomized algorithm for unsupervised manifold learning (Tenenbaum et al., 2000; Roweis and Saul, 2000; Donoho and Grimes, 2003; Belkin and Niyogi, 2003). The objective in manifold learning is to find a subspace or sub-manifold that captures local structure in the data which a global method such as PCA cannot. We focus on locality preserving projections (LPP) (He and Niyogi, 2003) since it can be efficiently applied to out-of-sample data. For all methods we will also provide adaptive algorithms to estimate the dimension of the projective linear subspace. In the next section we outline the main computational and statistical considerations underlying our proposed randomized approaches to dimension reduction for massive high-dimensional data.

2.1 Computational considerations

The main computational tools used are randomized approximate algorithms for matrix factorization. The key idea in randomized algorithms for matrix factorization is that an $n \times p$ matrix with low rank d can be approximately factored in time $\mathcal{O}(npd)$ rather than the $\mathcal{O}(n^2p)$ required for exact factorization. We assume that the data has low intrinsic dimensionality and hence $d \ll n < p$. Given the low rank factorization, matrix multiplications in the low dimensional space replace those in the ambient space with little loss in accuracy of the matrix operations. The randomized methods we will use provide flexible control of the degree of accuracy in the final approximation to the exact sample estimates and make explicit the tradeoff between computation time and estimation accuracy.

2.2 Statistical considerations

A central concept in this paper is that randomized approximation algorithms used for statistical inference impose regularization constraints. Thinking of the estimate computed by the randomized algorithm as a statistical model helps highlight this idea. Consider two models or estimators, one corresponding to the exact (generalized) eigendecomposition and the other corresponding to the randomized algorithm. The numerical analysis perspective typically focuses on the discrepancy between the randomized and the exact algorithm evaluated on the same data. The goal is to minimize the difference between the estimates, irrespective of the data generating process. However, in many practical applications the data is a noisy random sample from a population. This suggests, in particular when the interest is in dimension reduction, that the relevant error comparison should be between the true subspace that captures information about the population and the corresponding algorithmic estimates. A key parameter of the randomized estimators considered here is the number of power iterations used to estimate the span of the data. This parameter induces a solution path which converges to the solution of the exact method applied to the sample data as the number of iterations increases. Hence, the power iteration parameter controls both the computational tradeoff as well as the amount of regularization applied to the sample estimates. Fewer iterations correspond to stronger regularization and reduced runtime. We show in Section 3 that very high dimensional data requires a great deal of regularization to achieve small out-of-sample errors. This argues for very few iterations for reasons of both computational and statistical efficiency.

2.3 Notation

Given positive integers p and d with $p \gg d$, $\mathbb{R}^{p\times d}$ stands for the class of all matrices with real entries of dimension $p \times d$, and $SS^{p\times p}$ denotes the sub-class of symmetric positive semi-definite $p \times p$ matrices. For $B \in \mathbb{R}^{p\times d}$, span($B$) denotes the subspace of $\mathbb{R}^p$ spanned by the columns of B. A *basis matrix* for a subspace $\mathcal{S}$ is any full column rank matrix $B \in \mathbb{R}^{p\times d}$ such that $\mathcal{S} = \text{span}(B)$, where $d = \dim(\mathcal{S})$. Denote the data matrix by $X = (x_1, \ldots, x_n)^T \in \mathbb{R}^{n\times p}$, where each of the n samples x_i is a random draw from a p-dimensional probability distribution $\mathcal{P}_X$. In the case of supervised dimensions reduction, denote the response vector to be $Y \in \mathbb{R}^n$ (quantitative response) or $Y \in \{1, \ldots, r\}$ (categorical response with r categories), and $Y \sim \mathcal{P}_Y$. The data and the response have a joint distribution $(X, Y) \sim \mathcal{P}_{X\times Y}$. Unless explicitly specified otherwise, assume that both the sample data and the response (for the regression case) are centered, so that $\sum_{i=1}^{n} y_i = \sum_{i=1}^{n} x_{ij} = 0$ for all $j = 1, \ldots, p$.

2.4 Principal Component Analysis

In this section we provide a detailed description of the randomized algorithm used as an estimator for PCA. The randomization ideas we use draw heavily from previous work in randomized numerical analysis (Halko et al., 2009; Rokhlin et al., 2008) . Our main algorithmic contribution is based on the observation that the data is a noisy sample from a population quantity. This suggests that algorithmic considerations such as the number of iterations also impose modeling constraints such as how much regularization is needed to accurately estimate the true population subspace. The objective of PCA is given high-dimensional data find a set of $d \ll p$ linear combinations of the original p variables

$$\xi_j = b_j^T X, \quad j = 1, 2, \ldots, d,$$

which retain as much as possible of the variation in the original data set. The j-th coefficient vector $b_j = (b_{1j}, \ldots, b_{pj})$ satisfies the following

1. the linear projections $\xi_j,\ j = 1 \ldots, d$ are ordered by decreasing variance: $\text{var}(\xi_1) \geq \ldots \geq \text{var}(\xi_d)$;
2. ξ_i is uncorrelated with ξ_j for all $i \neq j$.

This problem is stably and efficiently solved, for an arbitrary rectangular matrix $X \in \mathbb{R}^{n \times p}$ and all possible values $d \in \{1, \ldots, n\}$, by *Singular Value Decomposition* (SVD):

$$X = USV^T, \quad U^T U = V^T V = I_{n \times n}, \tag{1}$$

$$S = \text{diag}(\ell_1, \ldots \ell_n), \ \ \ell_1 \geq \ell_2 \geq, \ldots \geq \ell_n \geq 0. \tag{2}$$

The diagonal entries of S are the singular values, sorted in non-decreasing order, according to the the amount of captured variance in the directions defined by the corresponding singular vectors. The first d orthonormal columns of the matrix V (with the d largest singular values), provide the exact solution

$B \in \mathbb{R}^{p \times d}$

$$B = \begin{pmatrix} | & & | \\ u_1 & \dots & u_d \\ | & & | \end{pmatrix}. \tag{3}$$

A numerically stable and computationally efficient implementation of SVD is provided as part of the industry-standard numerical linear algebra package LAPACK (Anderson et al., 1990) requiring $O(n^2p)$ time and $O(np)$ of memory. When both the number of variables (p) and the number of data points (n) is large this runtime cost is prohibitively high. In the next Section we describe how randomized algorithms address this problem.

2.4.1 Randomized PCA

It is possible to estimate the maximum variance directions, or principal components, without computing the full eigendecomposition of the covariance matrix. For example when the covariance matrix is low rank. In this case an attractive alternative would be a computationally accurate and efficient approximation of the top few eigenvector directions. Randomized matrix factorizations achieve such an approximation by coupling random projections with numerically stable factorizations of the resulting lower dimensional matrix objects. The idea of random projection was first developed as a proof technique to study the distortion induced by low dimensional embeddings of high-dimensional vectors in the work of Johnson and Lindenstrauss (1984) with much literature simplifying and sharpening the results (Frankl and Maehara, 1987; Indyk and Motwani, 1998; Dasgupta and Gupta, 2003; Achlioptas, 2001). More recently, the theoretical computer science and the numerical analysis communities discovered that random projections can be used for efficient approximation algorithms for a variety of applications (Drineas et al., 2006; Sarlos, 2006; Liberty et al., 2007; Boutsidis et al., 2009; Rokhlin et al., 2008; Halko et al., 2009). Given data $X \in \mathbb{R}^{n \times p}$ the following algorithm *Randomized SVD* computes an approximation of the top d singular values and singular vectors of X (Rokhlin et al., 2008; Halko et al., 2009). The algorithm proceeds in two stages: (1) efficiently identify an orthonormal basis for the range of X that captures the directions of high variation, (2) project the data onto this bases and apply SVD.

Algorithm: *Randomized SVD*

(1) Find orthonormal basis for the range of X;

   (i) Set the number directions: $\ell = d + \Delta$;

   (ii) Generate random matrix: $\Omega \in \mathbb{R}^{n\times\ell}$ with $\Omega_{ij} \overset{iid}{\sim} \mathrm{N}(0,1)$;

   (iii) Construct blocks: $F^{(i)} = XX^T F^{(i-1)}$ with $F^{(1)} = XX^T\Omega$ for $i = 1, .., \ell$;

   (iv) Factorize blocks: $R = (F^{(1)} \mid F^{(2)} \mid \ldots \mid F^{(t)}) = QS, \quad Q^TQ = I$;

(2) Project data onto basis and compute SVD;

   (i) Project onto basis: $B = X^TQ \in \mathbb{R}^{p\times(t\ell)}$;

   (ii) Compute SVD: $B = U\Sigma W^T, \quad \Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_{t\times\ell}), \quad U^TU = W^TW = I$;

   (iii) Compute eigenvectors: $\hat{U} = \begin{pmatrix} u_1 & \ldots & u_d \end{pmatrix}, \quad \hat{V} = QW, \quad \hat{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_d)$.

In stage (1) we first set the number of bases we expect to need ℓ. The parameter Δ is an oversampling parameter, we use 12 as suggested in Rokhlin et al. (2008). The parameter d is our expectation of the rank of the data matrix. We will state an adaptive procedure to estimate d later in this subsection. In step (iii) the random projection matrix Ω is applied to powers of XX^T to randomly sample linear combinations of singular vectors of the data weighted by powers of the singular values

$$\underbrace{F^{(i)}}_{n\times\ell} = (XX^T)^i\Omega = US^{2i}V^T\Omega = US^{2i}\Omega^*, \quad \text{where } X = USV^T.$$

The main goal of the power iterations is to increase the decay of the noise portion of the eigen-spectrum while leaving the singular vectors unchanged. This is a regularization or shrinkage constraint. Note that each column j of $F^{(i)}$ corresponds to a draw from a n-dimensional Gaussian distribution: $F_j^{(i)} \sim N(0, US^{4i}U^T)$, with the covariance structure more strongly biased towards higher directions of variation as i increases. This type of regularization is analogous to the local shrinkage term developed in Polson and Scott (2010). In step (iv) we combine blocks $F^{(i)}$ for $i = 1, .., t$ and factorize the collection using a QR factorization. Rokhlin et al. (2008) suggested the use of the collection to provide numerical stability in same spirit as the (blocked) Lanzcos approach (Chapter 9, in Golub and Van Loan (1996)). In stage (2) we rotate the orthogonal basis Q computed in stage (1) to the canonical eigenvector bases and scale according to the corresponding singular values. In step (i) the data is projected onto the low dimensional orthogonal basis Q. Step (ii) computes the singular values and vectors of the projected space. The computational complexity of the randomization step is $\mathcal{O}(t\ell np)$ where $\ell \approx d$ is the rank of the approximation and t is the number of iterations of the (blocked) Lanzcos step. The factorization step is $\mathcal{O}(\ell np + t^2\ell^2 n)$. With t and ℓ very small relative to n, p the resulting overall complexity is $O(t\ell np + t^2\ell^2 n)$. The runtime in both steps is dominated by the multiplication by the data matrix and in the case of sparse data fast multiplication can further reduce the runtime. We use a normalized version of the above algorithm that has the same runtime complexity but is numerically more stable (Martinsson et al., 2010). The exact algorithm follows:

Algorithm 1 : Randomized SVD

```
input: $X \in \mathbb{R}^{n\times p}$: data matrix, $d$: number of required top eigenvalue directions, $\Delta$: oversampling parameter, $t$: number of power iterations
output: $\hat{\Sigma} \in \mathbb{R}^{d}$: top singular values, $\hat{U} \in \mathbb{R}^{p\times d}$: left eigenvectors, $\hat{V} \in \mathbb{R}^{n\times d}$: right eigenvectors
Stage 1: Find approximate orthonormal basis for the range of $X$
  1. Set $l = d + \Delta$
  2. Generate $\Omega \in \mathbb{R}^{n\times l}$ s.t. $\Omega_{ij} \sim N(0,1) \quad \forall i, j$
  3. Set $F^{(0)} = \Omega$. for $j = 1, \dots, t$
     (a) Set $F^{(j)} = X^T F^{(j-1)}$
     (b) Factorize $F^{(j)} = Q_1^{(j)} S_1^{(j)}, \quad Q_1^{(j)T} Q_1^{(j)} = \mathrm{I}$
     (c) Set $F^{(j)} = X Q_1^{(j)}$
     (d) Factorize $F^{(j)} = Q_2^{(j)T} S_2^{(j)}, \quad Q_2^{(j)} Q_2^{(j)} = \mathrm{I}$
     (e) Set $F^{(j)} = Q_2^{(j)}$
  4. Factorize $R = (F^{(1)} \mid F^{(2)} \mid \dots \mid F^{(t)}) = QS \in \mathbb{R}^{n\times tl}, \quad Q^T Q = \mathrm{I}$
Stage 2: Project data onto the orthonormal basis $Q$ and do SVD
  1. Project & Factorize $[\tilde{U}, \tilde{\Sigma}, \tilde{V}] =$ full SVD$(Q^T X)$ [LAPACK]
  2. Set $\hat{\Sigma} = \mathrm{diag}(\tilde{\sigma}_1, \dots, \tilde{\sigma}_d), \quad \hat{V} = \tilde{V}[\,, 1:d], \quad \hat{U} = Q\tilde{U}[\,, 1:d]$
```

2.4.2 Adaptive method to estimate d

In the unsupervised setting, for PCA and LPP, we use the expected reconstruction accuracy upon orthogonal projection onto the inferred dimension reduction subspace of a held-out subset of the data as the criterion to select the true rank of the data d^*. Optimizing this criterion in the population corresponds to

$$d^* = \underset{d \in \{1,\ldots,d_{max}\}}{\arg\min} \; \mathbb{E}_X \| (I - P_{\tilde{B}^{(d)}}) X \|_2^2 \equiv \| (I - \tilde{B}^{(d)}_{(-i)} \tilde{B}^{(d)T}_{(-i)}) X \|_2^2.$$

Leave-one-out Cross-validation. Leave-one-out cross-validation (LOO-CV) provides an approximately unbiased estimate of d^* based on the random sample $\{X_i\}_{i=1}^n$. First, for each d a loss function based on predictive performance on held out data is defined as

$$\mathrm{CV}^{(d)}_{\text{loo}} = \frac{1}{n} \sum_{i=1}^{n} \| (I - P_{\tilde{B}^{(d)}_{(-i)}}) X_i \|_2^2 = \frac{1}{n} \sum_{i=1}^{n} \| (I - \tilde{B}^{(d)}_{(-i)} \tilde{B}^{(d)T}_{(-i)}) X_i \|_2^2,$$

where $\tilde{B}^{(d)}_{(-i)} \in \mathbb{R}^{p \times d}$ is a basis matrix for the dimension reduction subspace inferred using $\{X_{(-i)}\}$, the full data set excluding the i-th sample.

C-fold Cross-validation. A more computationally efficient estimate that has lower variance but higher bias than LOO-CV is provided by c-fold cross-validation (e.g. c = 5 or 10). To simplify notation let $n = a \times c$ and denote the c-fold cross-validation loss function estimate for a pre-specified value of the parameter d to be $\mathrm{CV}^{(d)}_{\text{c-fold}}$. Randomly partition the input data $\{X_i\}_{i=1}^n$ into c equally sized disjoint subsets $\{X_{(i)}\}_{i=1}^c$, where $X^j_{(i)}$ denotes the j-th sample point within subset i. Then

$$\mathrm{CV}^{(d)}_{\text{c-fold}} = \frac{1}{c} \sum_{i=1}^{c} \frac{1}{a} \sum_{j=1}^{a} || (I - P_{\tilde{B}^{(d)}_{(-i)}}) X^j_{(i)} ||_2^2 = \frac{1}{n} \sum_{i=1}^{c} \sum_{j=1}^{a} || (I - \tilde{B}^{(d)}_{(-i)} \tilde{B}^{(d)T}_{(-i)}) X^j_{(i)} ||_2^2,$$

where $\tilde{B}^{(d)}_{(-i)} \in \mathbb{R}^{p \times d}$ is the orthogonal basis matrix for the dimension reduction subspace inferred using the full data set excluding the i-th subset of size a, $X_{(i)}$. We optimize over the full range of allowable values for d to arrive at the final estimate

$$\hat{d}^*_{\text{c-fold}} = \underset{d \in \{1,\ldots,d_{max}\}}{\arg\min} \; \mathrm{CV}^{(d)}_{\text{c-fold}}.$$

Runtime Analysis. The overall asymptotic runtime scales linearly with c for a fixed value of the parameter k. In more detail, for a fixed value of of $d \in \{1, \ldots, d_{\max}\}$ there are two majors computational steps:

- estimate projective subspace $\tilde{B}^{(d)}_{(-i)}$, for $i = 1, \ldots, c$: $O(c \times [tl(n-a)p + t^2 l^2 (n-a)])$
- project data and construct error estimate: $O(lnp)$

Hence, the overall asymptotic runtime is dominated by the projective subspace estimation and is of $O(d_{\max} \times c \times [tl(n-a)p + t^2 l^2 (n-a)])$. For small values of $d_{\max}$ this is on the order of a single iteration of Algorithm 1.

2.4.3 Accuracy of the randomized algorithm

Assumig the the data matrix is a noisy random draw from the population quantity there are main two criteria by which the accuracy of the approximate algorithm can be measured. The first is an estimate of the accuracy of the approximate factorization on the observed data matrix. The second criteria is the error between the estimated subspace and the subspace that captures the structure of the population model. We estimate the second criterion using held-out subset of the data which is not used for estimation. In terms of the error in reconstructing the data matrix it has been shown (Rokhlin et al., 2008; Halko et al., 2009) that the randomized SVD algorithm is highly accurate and computationally efficient. In Rokhlin et al. (2008) the following bound on the reconstruction error of the rank-d approximation to X was given

$$||X - \hat{U}\hat{\Sigma}\hat{V}^T||_2 \leq C n^{1/(4t)} \sigma_{d+1},$$

where $\sigma_{(d+1)}$ is the $(d+1)^{st}$ singular value of X, $|| \cdot ||_2$ denotes the spectral norm, t is the number of power iterations, and C is a constant independent of X. In addition, Rokhlin et al. (2008) demonstrated through numerical experiments that the theoretical bounds hold in practice with greatly reduced scaling constants as compared to the theoretical predictions, independent of the matrix being approximated. From an inference perspective a factorization that exactly reconstruct the data matrix X may not be ideal because it may be susceptible to overfitting the noise in the data, especially when the variable dimension is high relative to the sample size. Hence, rather than optimizing the exact reconstruction accuracy we propose to optimize the tradeoff between exactly fitting the sample data and good performance on future data. This strategy is implemented using an out-of-sample predictive accuracy criterion to set an appropriate value for the regularization parameter t. We will show in Section 3 different scenarios when the dimension reduction estimators have better out-of-sample performance for smaller values of t (large shrinkage) than for large t

(little shrinkage). In short, poor estimators for the in-sample data matrix can be superior on out-of-sample data.

2.5 Generalized eigendecomposition and dimension reduction

For each of the other three dimension reduction methods: SIR, LSIR, and LPP, we will provide randomized approximate algorithms to address the problems of supervised and unsupervised dimension reduction capturing non-linear (manifold) structure. Inference for all three methods involves the solution of a generalized eigedecomposition problem of the following form

$$\Gamma g = \lambda \Sigma g, \tag{4}$$

where $\Sigma, \Gamma \in SS^{p\times p}$ are symmetric positive semi-definite matrices. Let $d \ll \min(n, p)$ be the intrinsic dimensionality of information contained in the data. Then our main objective is to find a basis for the subspace spanned by the generalized eigenvectors $\{g_1^*, \ldots, g_d^*\}$. An important structural constraint in all three problems is that Γ is low-rank. It is this constraint that we will take advantage of in the randomized methods.

2.5.1 Supervised dimension reduction

Dimension reduction is often a first step in further downstream data analysis including data visualization and regression. If the ultimate goal is regression then the low dimensional summary $Z \equiv R(X)$ is computed from the data for which there exists an accurate predictive model of the future response variables

$$Y = f(X) + \varepsilon = h(Z) + \varepsilon, \quad X \in \mathbb{R}^p, Z \in \mathbb{R}^d, \; d \ll p.$$

Sufficient dimension reduction (SDR) (Li, 1991; Cook and Weisberg, 1991; Li, 1992; Hastie and Tibshirani, 1996; Goldberger et al., 2005; Globerson and Roweis, 2006; Li et al., 2005; Nilsson et al., 2007; Sugiyama, 2007; Cook, 2007; Wu et al., 2010) aims to provide a summary $R(X)$, which captures the predictive information of X about the response Y. A very useful consequence for predictive modeling is the identity $\mathbb{E}[Y \mid X] = \mathbb{E}[Y \mid R(X)]$, see Cook (2007) for more details. In this paper we focus on linear SDRs which correspond to projections of the data onto a subspace $G = (g_1, \ldots, g_d) \in \mathbb{R}^{p\times d}$ called the dimension reduction subspace. Hence, the following condition will hold:

$$(Y \mid X) \overset{d}{=} (Y \mid G^T X), \quad \overset{d}{=} \text{ is equivalence in distribution.}$$

We will consider two specific supervised dimension reduction methods: Sliced Inverse Regression (SIR) (Li, 1991) and Localized Sliced Inverse Regression (LSIR) (Wu et al., 2010). SIR is effective when the predictive structure in the data is global, i.e. there is single predictive subspace over the support of the marginal distribution of X. In the case of local or manifold predictive structure in the data, LSIR can be used to compute a projection matrix G that contains this non-linear (manifold) structure.

Inverse Regression (SIR) Sliced Inverse Regression (SIR) is a dimension reduction approach introduced by Li (1991). The relevant statistical quantities for SIR are the covariance matrix, $\Sigma = \text{cov}(X)$, and the covariance of the inverse regression, $\Gamma = \text{cov}[\mathbb{E}_X(X|Y)]$. We define the following eigendecomposition problem

$$\Gamma g_i = \lambda_i \Sigma g_i, \tag{5}$$

and the dimension reduction subspace is $G = \text{span}(g_1, ..., g_d) \mid \lambda_1, ..., \lambda_d > 0$, the eigenvectors corresponding to non-zero eigenvalues. If there exists a linear projection matrix G with the property that

$$\text{span}(\mathbb{E}_X[X \mid Y] - \mathbb{E}_X[X]) \in \text{span}(\Sigma G) \tag{6}$$

then SIR is effective. Given observations $\{(x_i, y_i)\}_{i=1}^n$ we obtain an estimate of $\hat{G}$ by computing empirical estimates $\hat{\Sigma}$ and $\hat{\Gamma}$. A standard estimator for the covariance is used $\hat{\Sigma} = \frac{1}{n}\sum_{i=1}^n x_i x_i^T$. To compute $\hat{\Gamma}$ the data is first divided into H groups or slices S_h with n_h observations in each group and then $\hat{\Gamma}$ is computed

$$\hat{\Gamma} = \sum_{h=1}^{H} \frac{n_h}{n} \hat{\mu}_h \hat{\mu_h}^T, \quad \hat{\mu}_h = \frac{1}{n_h} \sum_{j \in S_h} x_j,\ h = 1, ..., H.$$

The empirical version of (5) is solved to compute $\hat{G}$

$$\hat{\Gamma} g_i = \lambda_i \hat{\Sigma} g_i, \quad \hat{G} = \text{span}(g_1, ..., g_d) \mid \lambda_1, ..., \lambda_d > \delta, \tag{7}$$

where $\delta > 0$ is a fixed threshold. Problems with SIR arise when the statistical assumption in (6) is not satisfied, that is the predictive structure in the data is non-linear. This can be due to the data being concentrated near a manifold or having clustering structure. SIR can be adapted to address this setting by taking into account the *local* structure of the explanatory variables conditioned on the response variable. The key idea behind Localized Sliced Inverse Regression (LSIR) (Wu et al., 2010) is based on the observation in manifold learning that Euclidean structure around a data point in $\mathbb{R}^p$ is only useful locally. This suggests computing a local version of the covariance of the inverse regression $\hat{\Gamma}_{\text{loc}}$. First we compute a local mean for each observation:

$$\hat{\mu}_{i,\text{loc}} = \frac{1}{k} \sum_{j \in P_i} x_j,$$

where P_i denotes the set of indexes of the k-nearest neighbors, considering only the data points with response values within the slice to which x_i belongs. Given the local means we compute $\hat{\Gamma}_{\text{loc}}$ as

$$\hat{\Gamma}_{\text{loc}} = \frac{1}{n} \sum_{i=1}^{n} \hat{\mu}_{i,\text{loc}}\, \hat{\mu}_{i,\text{loc}}^T$$

and solve the generalized eigendecomposition problem

$$\hat{\Gamma}_{\text{loc}}\, g_i = \lambda_i \hat{\Sigma} g_i, \quad \hat{G} = \text{span}(g_1, ..., g_d) \mid \lambda_1, ..., \lambda_d > \delta. \tag{8}$$

For both SIR and LSIR the most straightforward solution to the generalized eigendecompositions in equations (7) and (8) is based on the Cholesky decomposition of $\hat{\Sigma}$ and typically requires runtime complexity O(np^2). In the next subsection we provide an efficient algorithm to compute the generalized eigendecompositions based on the (approximate) SVD of the data matrix X.

Efficient solutions and approximate SVD Both SIR and LSIR reduce to solving a truncated generalized eigendecomposition problem

$$\hat{\Gamma} g = \lambda \hat{\Sigma} g, \tag{9}$$

with $\hat{G}$ as the subspace spanned by the generalized eigenvectors $g_1, \ldots, g_d$, with largest generalized eigenvalues $\lambda_1, \ldots, \lambda_d$, and d is an estimate of the intrinsic dimensionality of the data – the number of predictive directions. We propose an alternative to the full Cholesky-based factorization that takes advantage of the low-rank structure in $\hat{\Gamma}$. Set $\hat{\Gamma}^{\frac{1}{2}} = SU^T$ as the square root of the spectral factorization $\hat{\Gamma} = US^2U^T = (\hat{\Gamma}^{\frac{1}{2}})^T \hat{\Gamma}^{\frac{1}{2}}$. The generalized eigendecomposition problem can be rewritten as the following symmetric eigendecomposition

$$(\hat{\Gamma}^{-\frac{1}{2}})^T \hat{\Sigma} \hat{\Gamma}^{-\frac{1}{2}} e = \frac{1}{\lambda} e, \qquad e \equiv \hat{\Gamma}^{\frac{1}{2}} g. \tag{10}$$

The dimension reduction subspace $\hat{G}$ will be span of $(g_1 = \hat{\Gamma}^{-\frac{1}{2}} e_1, \ldots, g_d = \hat{\Gamma}^{-\frac{1}{2}} e_d)$, and d is the rank of $\hat{\Gamma}$. If $\hat{\Sigma}$ and $\hat{\Gamma}$ can be computed efficiently and are (approximately) low-rank then we will show that the above computation can be much faster than the Cholesky-based factorization. We first consider the case of SIR. Sort the samples in decreasing order of the response and slice the samples into H slices. For each slice h we define the row vector $J_h = (1 \ \cdots \ 1)_{1 \times n_h}$ and the matrix

$$J_{\text{sir}} = \begin{pmatrix} J_1 & & 0 \\ & \ddots & \\ 0 & & J_H \end{pmatrix}_{n \times n}.$$

Given J_{sir} and the data matrix we observe that

$$\hat{\Gamma} = X^T J_{\text{sir}}^T J_{\text{sir}} X.$$

By construction $\text{rank}(\hat{\Gamma}) \leq H - 1 \ll \min(n, p)$ and $\hat{\Gamma} \equiv X^T J_{\text{sir}} \in \mathbb{R}^{p \times H}$ can be constructed in $O(np)$ time. The full eigendecomposition of $\hat{\Gamma}^{\frac{1}{2}}$ is $O(H^2 p)$. Recall that $X = \hat{\Sigma}^{1/2}$ so the explicit construction and full eigendecomposition of $\hat{\Sigma}^{\frac{1}{2}} \hat{\Gamma}^{-\frac{1}{2}} = XUS^{+}$ is $O(Hnp + H^2 p)$. Hence the computation of the exact solution to the optimization problem posed by SIR requires $O(H^2 p + Hnp)$ time. This can be made faster by using RSVD approach in Section 2.4.1. When the number of slices is small, $H \ll \min(n, p)$, the time savings could be substantial as compared to a typical Cholesky-based factorization which scales as $O(n^2 p)$. We now consider the LSIR case. For each slice $h \in \{1, \ldots, H\}$ we construct a block matrix J^h that is $n_h \times n_h$ with each entry $J^h_{ij} = \frac{1}{k}$ if i, j are k-nearest neighbors (k fixed parameter) and 0 otherwise, $J^h_{ii} = \frac{1}{k}$. We then construct the block diagonal $n \times n$ matrix $J_{\text{lsir}} = \text{diag}(J_1, ..., J_H)$. Then

$$\hat{\Gamma}_{\text{loc}} = X^T J_{\text{lsir}}^T J_{\text{lsir}} X.$$

To compute the k-nearest neighbors we need the pairwise distances between points. We do not do this in the p-dimensional data space but use the eigenvector coordinates of the marginal covariance matrix. Given the SVD of X constructing $\hat{\Gamma}^{\frac{1}{2}} := X^T J_{lsir} \in \mathbb{R}^{p \times n}$ requires $O(dnp)$ operations. Computing the SVD of X using the RSVD approach in Section 2.4.1 requires $O(t\ell np)$ operations. Similarly an approximate eigendecomposition of $\hat{\Gamma}^{\frac{1}{2}}$, $\hat{\Gamma}_*^{\frac{1}{2}}$, requires $O(t\ell np)$ operations. The final step of finding the top eigenvalues and eigenvectors of $\hat{\Gamma}_*^{-\frac{1}{2}} \hat{\Sigma}^{\frac{1}{2}} = \hat{\Gamma}_*^{-\frac{1}{2}} X^T \in \mathbb{R}^{d \times n}$ using exact SVD and back-transforming into the ambient basis takes $O(dnp)$. This brings the runtime complexity of the approximate solution to

$O((t\ell + d)np)$ versus$O(n^2p)$ for the standard Cholesky-based approximation. The detailed algorithm is stated in Algorithm 2.

Algorithm 2 : Randomized (L)SIR

```
input: $X \in \mathbb{R}^{n\times p}$: data matrix, $H$: number of slices, $k$: number of nearest neighbors [LSIR only], $d$: dimension of sufficient dimension reduction subspace, $\Delta$: oversampling parameter for *Ranomized SVD* [default: $\Delta = 12$], $t$: number of power iterations for *Randomized SVD* [default: t=1] output: $\hat{G}_{(L)SIR} \in \mathbb{R}^{p\times d}$: a basis matrix for the effective dimension reduction subspace Stage 1: Estimate low-rank approximation to $\Gamma$
    1. Construct $J_{(L)SIR}$, as described in Section 2.5.1
    2. Set $L = X^T J^T_{(L)SIR}$ $(\Rightarrow \hat{\Gamma}_{(L)SIR} = LL^T)$
    3. Factorize $[U, S, V] = \textit{RandomizedSVD}(L, d, \Delta, t)$
Stage 2: Solve the generalized eigendecomposition
    1. Construct $A = SU^T X^T \in \mathbb{R}^{d\times p}$
    2. Factorize $[\tilde{U}, \tilde{S}, \tilde{V}] =$ full SVD(A) [LAPACK]
    3. Back-transform $\hat{G}_{(L)SIR} = \tilde{U} S U^T \in \mathbb{R}^{p\times d}$
```

2.5.2 Adaptive method to estimate d

In the supervised setting, for SIR and LSIR, we use the expected prediction error from kNN regression/classification upon orthogonal projection onto the inferred dimension reduction subspace as the criterion to select the true rank of G, d^*. In particular, for regression we use the expected squared prediction error estimated by the sums of squares of the predicted residuals as the optimization criterion while for classification we select the value for d^* that minimizes the expected misclassification accuracy as estimated by the observed misclassification performance. Optimizing this criterion in the population corresponds to

$$d^* = \underset{d\in\{1,\ldots,d_{\max}\}}{\arg\min} \; \mathbb{E}_{Y,X} \| L(Y, \hat{Y}^{(d,k)}) \|,$$

where $\hat{Y}^{(d,k)} = \text{kNN}(X\hat{G}^{(d)}, Y, k)$ and $L(Y, \hat{Y}^{(d,k)}) = ||Y - \hat{Y}^{(d,k)}||_2^2$ for regression and $L(Y, \hat{Y}^{(d,k)}) = 1_{Y \neq \hat{Y}^{(d,k)}}$ for classification. Here, k is a fixed parameter denoting the number of nearest neighbors used for the class estimation and $\hat{G}^{(d)}$ is the estimate of a basis matrix for the edr subspace of dimension d.

Leave-one-out Cross-validation. Similarly to Section 2.4.2, we propose to use the Leave-one-out cross-validation (LOO-CV) estimator for d^*, based on the random sample $\{X_i\}_{i=1}^n$. First, for each d the expected loss is estimate using held out data

$$\text{CV}_{\text{loo}}^{(d)} = \frac{1}{n}\sum_{i=1}^{n} L(Y_i, \hat{Y}_{(-i)}^{(d,k)}),$$

where $\hat{Y}_{(-i)}^{(d,k)} = \text{kNN}(X\hat{G}_{(-i)}^{(d)}, Y, k)$ is the k nearest neighbor prediction for Y_i, based on projection of the data onto estimated reduction subspace inferred using $\{Y_{(-i)}, X_{(-i)}\}$, the full data set excluding the i-th sample.

C-fold Cross-validation. In practice, for massive data it is of importance to use a more computationally efficient estimate provided by c-fold cross-validation (e.g. c = 5 or 10). To simplify notation let $n = a \times c$ and denote the c-fold cross-validation loss function estimate for a pre-specified value of the parameter d to be $\text{CV}_{\text{c-fold}}^{(d)}$. Randomly partition the input data $\{(X_i, Y_i)\}_{i=1}^n$ into c equally sized disjoint subsets $\{D_i := (X_{(i)}, Y_{(i)})\}_{i=1}^c$, where $\{(X_i, Y_i)\}_{i=1}^n = \cup_{i=1}^c D_i$. Then

$$\text{CV}_{\text{c-fold}}^{(d)} = \frac{1}{c}\sum_{i=1}^{c}\frac{1}{a}\sum_{j=1}^{a} L(Y_{i,j}, \hat{Y}_{(-i,j)}^{(d,k)}),$$

where Y_{ij} denotes the observed response for the j-th sample within class i and $\hat{Y}_{(-i,j)}^{(d,k)}$ is the predicted value for the j-th sample point in the i-th subset D_i, based on projection of the data onto the estimated reduction subspace. The inference is based on the full data set, excluding the i-th subset. We optimize over the full range of allowable values for d to arrive at the final estimate

$$\hat{d}_{\text{c-fold}}^{*} = \underset{d \in \{1, \ldots, d_{max}\}}{\arg\min} \text{CV}_{\text{c-fold}}^{(d)}.$$

2.5.3 Unsupervised manifold learning

Dimension reduction based on graph embeddings seek to map the original data points to a lower dimensional set of points while preserving neighborhood relationships. The theoretical assumptions underlying these methods are that the data lies on a smooth manifold embedded in the high-dimensional ambient space. This manifold is unknown and needs to be inferred from the data. Given sufficient number of observations the manifold can be reasonably represented as a $G = (E, V)$ (Chung, 1997) where the vertexes $\{v_1, .., v_n\}$ correspond to the n observations $\{x_1, .., x_n\}$ and the edges corresponds to which points which are close to each other. For example, this neighborhood relationship can be encoded in a sparse symmetric adjacency matrix W. Given W, the Laplacian Eigenmaps algorithm Belkin and Niyogi (2003) embeds the data into a low dimensional space preserving local relationships between points. Given the adjacency or association matrix the Graph Laplacian is constructed $L = D - W$, where D is a diagonal matrix with $D_{ii} = \sum_j W_{ij}$. A spectral decomposition of L

$$Lv_i = \lambda_i v_i$$

results in eigenvalues $\lambda_1 = 0 \leq \lambda_2 \leq ... \leq \lambda_n$ with $v_1 = \mathbf{1}$. Projecting the matrix L onto the d eigenvectors corresponding to the smallest d eigenvalues greater than zero embeds the n points into a d dimensional space. Under certain conditions (Belkin and Niyogi, 2005), the Graph Laplacian converges to the Laplace-Beltrami operator on the underlying manifold. This provides a theoretical motivation for the embedding. This embedding needs to be recomputed when a new data point is introduced and typically will not be a linear projection of the data. Hence, for computational reasons it would be advantageous to have a linear projection that can be applied to new data points without having to recompute the spectral decomposition of the graph Laplacian. The goal of Locality Preserving Projections (He and Niyogi, 2003) is to provide such a linear approximation to the non-linear embedding of Laplacian Eigenmaps (Belkin and Niyogi, 2003). The dimension reduction procedure starts by specifying the dimension of the transformed space to be $d < n$. Let the parameter defining the neighborhood size be k. Locality Preserving Projections (LPP) (He and Niyogi, 2003) is stated as the following generalized eigendecomposition problem.

$$\begin{aligned} X^T L X e &= \lambda X^T D X e, \\ X^T (D - W) X e &= \lambda X^T D X e, \\ X^T W X e &= (1 - \lambda) X^T D X e, \\ \tilde{X}^T \tilde{W} \tilde{X} e &= \mu \tilde{X}^T \tilde{X} e, \end{aligned} \tag{11}$$

where, $\tilde{X} = D^{\frac{1}{2}} X$, $\tilde{W} = D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$, $\mu = 1 - \lambda$ and for a fixed bandwidth parameter b,

$$W_{ij} = \begin{cases} \exp\left\{ -\frac{||X_i - X_j||}{\text{b}} \right\} & \text{if } X_i \text{ is among the } k\text{-NN of } X_j \text{ or vice versa} \\ 0 & \text{otherwise.} \end{cases}$$

The column vectors that are the solutions to equation (11) are the required embedding directions $\{e_j\}$, ordered according to their generalized eigenvalues $1 = \mu_0 > \mu_1 > \ldots > \mu_{d-1}$. Hence the neighborhood-preserving optimal embedding according to the LPP criterion is:

$$x_i \to A^T y_i, \quad \text{where } A = (e_0, e_1, \ldots, e_{d-1}).$$

LPP is obtained from a graph embedding, using a nearest neighbor graph that captures *local* manifold structure. If a complete graph is generated using Euclidean inner products between data points, then a very similar result to PCA is produced, emphasizing the *global* structure of the data ($W = XX^T$).

$$\begin{aligned} X^T W X e &= \mu X^T D X e \\ (X^T X)^2 e &= \mu X^T D X e. \end{aligned}$$

In the case when the diagonal matrix D is close to identity matrix, $X^T D X \approx X^T X$ the minimum eigenvalues of equation (12) correspond to the maximum eigenvalues of

$$\begin{aligned} (X^T X)^2 e &= \mu X^T X e \\ X^T X e &= \mu e, \end{aligned}$$

which is the optimization problem that is solved by PCA. Hence, LPP with a complete inner product graph is similar to PCA. The only difference is that the matrix D is used to measure the local density around each data point (by its degree on the neighborhood graph) while PCA treats all point equally.

Randomized Algorithm for LPP The approximation algorithm to solve LPP requires solving the generalized eigendecomposition stated by the last equation in derivation 11. There will be on the order of kn non-zero entries. Hence, the matrix $\tilde{W}$ will be sparse, assuming the size of the local neighborhoods $k \ll n$. This implies that the matrix product $X^T W X \Omega$, where $\Omega \in \mathbb{R}^{p \times l}$ can be efficiently computed in $O((k + l)np)$ time, without explicitly constructing W by using the k-nearest neighbor matrix. Hence

$\tilde{\Gamma} := \tilde{X}^T\tilde{W}\tilde{X}$ can be efficiently approximated using RSVD from Section 2.4.1 which would provide $\tilde{\Gamma}^{\frac{1}{2}} = SU^T$. Then, setting $\tilde{\Sigma} := \tilde{X}^T\tilde{X}$, we can proceed analogously to Section 2.5.1 to solve the problem

$$(\tilde{\Gamma}^{-\frac{1}{2}})^T\tilde{\Sigma}\tilde{\Gamma}^{-\frac{1}{2}}e = \frac{1}{\lambda}e, \qquad e \equiv \tilde{\Gamma}^{\frac{1}{2}}g. \tag{12}$$

The dimension reduction subspace $\hat{G}$ will be span of $(g_1 = \tilde{\Gamma}^{-\frac{1}{2}}e_1, \ldots, g_d = \tilde{\Gamma}^{-\frac{1}{2}}e_d)$. The rank d of $\tilde{\Gamma}$ and can be efficiently estimated using the adaptive procedure stated in Section 2.4.2.

Algorithm 3 : Randomized LPP

input: $X \in \mathbb{R}^{n\times p}$: data matrix, k: number of nearest neighbors capturing local manifold structure, d: dimension of the embedding subspace, Δ: oversampling parameter for *Randomized SVD* [default: $\Delta = 12$], t: number of power iterations for *Randomized SVD* [default: t=1]

output: $\hat{G}_{LPP} \in \mathbb{R}^{p\times d}$: a basis matrix for the embedding subspace **Stage 1**: Estimate low-rank approximation to $\tilde{X}^T\tilde{W}\tilde{X}$

```
1. Construct $\tilde{X}$ and $\tilde{W}$,
2. Factorize $[U, S, U] = \textit{RandomizedSVD}(\tilde{X}^T\tilde{W}\tilde{X}, d, \Delta, t)$
```

Stage 2: Solve the generalized eigendecomposition

```
1. Construct $A = SU^T\tilde{X}^T \in \mathbb{R}^{d\times p}$
2. Factorize $[\tilde{U}, \tilde{S}, \tilde{V}] =$ full SVD(A) [LAPACK]
3. Back-transform $\hat{G}_{LPP} = \tilde{U}SU^T \in \mathbb{R}^{p\times d}$
```

3 Results on real and simulated data

We use real and simulated data to demonstrate two points: (1) in both the supervised as well as unsupervised setting the randomized algorithms are much faster than the exact methods with minimal loss in estimation accuracy and (2) in the supervised setting the randomized algorithms are both much more computationally efficient and can have better out-of-sample accuracy than the exact methods. The second point is shown to be true in the *least eigenvalue setting*, when the response is associated strongly only with the least important principal component (Hotelling, 1957; Cox, 1968; Jolliffe, 1982).

3.1 Simulations

3.1.1 Unsupervised dimension reduction

We begin with unsupervised dimension reduction and focus on PCA comparing an exact version of SVD with the randomized SVD method introduced in Section 2.4. The simulated data will be drawn from a model with spiked Wishart covariance structure:

$$X_i \sim \mathrm{N}(0, \mathrm{diag}(\sigma_1, \ldots, \sigma_p)), \quad \text{for } i = 1, \ldots, n,$$

where we will vary the decay in the eigenvalues in the simulations. Our objective is to characterize the runtime advantage of the randomized SVD and to assess the relative sample eigenvalue estimation accuracy as a function of the number of iterations t. We first explore the effect of the regularization parameter t – the number of iterations of the power method. We spike the first twenty eigenvalues , $\sigma_j = 5/\sqrt{n}$, $1 \leq j \leq 20$, and then model the remaining eigenvalues as noise, $\sigma_j = 1/\sqrt{n}$, $20 < j \leq p$. We set the oversampling parameter $\Delta = 10$ and let t range from 1 to 5. For an input matrix of dimension $n = 2000$ and $p = 1000$ we plot the percent relative accuracy over 10 draws in Table 1. Using the same simulation setting as before

t	1	2	3	4	5
% relative accuracy	73.69 ± 0.44	91.01 ± 0.39	97.57 ± 0.41	99.44 ± 0.18	99.88 ± 0.05

Table 1: *Randomized SVD* eigenvalue estimation accuracy for different levels of regularization t.

and fixing the regularization parameter to be $t = 3$ we computed the ratio of the runtime of the exact

SVD over the randomized SVD: $r = \frac{\text{exact SVD}}{\text{randomized SVD}}$. Based on 10 random data sets we observe that $r = \{12.43 \pm 0.14, 17.13 \pm 0.11, 24.99 \pm 0.27\}$ for $p = \{1000,\ 1500,\ 2000\}$ respectively. Notice the gain increases as the matrix becomes more square – the smaller dimension approaches the larger dimension as is the case for massive high-dimensional data. To study the comparison of the exact and randomized SVD when there are two eigen-gaps we set the eigenvalues to be $\sigma_j = 10/\sqrt{n}$, $1 \leq j \leq 10$, $\sigma_j = 5/\sqrt{n}$, $10 < j \leq 20$, and $\sigma_j = 1/\sqrt{n}$, $20 < j \leq p$. The sample size and the dimension are is fixed to be $n = 2000, p = 1000$. We fix the oversampling $\Delta = 10$ and the number of iterations $t = 2$. Over 10 draws the relative estimation accuracy were 99.88 ± 0.02 for the eigenvalues above the first gap, 98.34 ± 0.24 for the eigenvalues between the first and second gap, and 93.06 ± 0.27 for the eigenvalues below the second gap. The eigenvalue estimates rapidly converge for the top two regimes.

3.1.2 Supervised dimension reduction of a latent factor regression model

In this section we explore the accuracy of randomized dimension reduction in the setting of factor regression. The modeling assumption is that the response variable Y is strongly correlated with a few directions in covariate space X. We compare six methods for estimation of the predictive subspace: SIR as implemented in Weisberg (2002) is denoted as dr. sir, SIR as implement by the (randomized) Algorithm 2 is denoted as rand.sir, LSIR as implemented in Wu et al. (2010) is denoted as lsir, LSIR as implemented by the (randomized) Algorithm 2 is denoted as rand.lsir, PCA denoted pca, and (randomized) PCA as implemented by the (randomized) Algorithm 1 (rand.pca). We will use a latent factor regression model (West, 2003) to simulate the data used to evaluate the accuracy and speed of the six methods. The rationale behind using a latent factor regression model is that we have explicit control of the direction of variation in covariates space X which is strongly correlated with the response Y. The latent factor model corresponds to the following decomposition

$$\begin{aligned} Y_i &= \lambda_i^T \theta + \epsilon_i, \\ X_i &= B S \lambda_i + \nu_i, \end{aligned}$$

with $d \ll p$, $B \in \mathbb{R}^{p \times d}$, $B^T B = I_d$, and $S = \text{diag}(s_1, \ldots, s_d)$. The random variables $\lambda_i, \nu_i, \epsilon_i$ are assumed to be independent and normally distributed

$$\begin{pmatrix} \lambda_i \\ \epsilon_i \\ \nu_i \end{pmatrix} \sim \text{N}(0, V), \;\; V = \begin{pmatrix} I_d & 0 & 0 \\ 0 & \tau^2 & 0 \\ 0 & 0 & \Delta^2 \end{pmatrix}.$$

Under the above model as the noise in the covariates decreases (Δ goes to zero) we obtain

$$Y_i \mid x_i \quad \sim \quad \text{N}(x_i^T B S^{-1} \theta, \tau^2),$$

which corresponds to the principal components regression model

$$Y_i \mid x_i \sim \text{N}(x_i^* \theta^*, \tau^2), \quad \text{with } x_i^* = x_i B \in \mathbb{R}^d \text{ and } \theta^* = S^{-1}\theta.$$

The dependence between X_i and Y_i is induced by marginalizing the latent factors λ_i. The joint distribution for (Y_i, X_i) is normal

$$\begin{pmatrix} X_i \\ Y_i \end{pmatrix} \sim \text{N}(0, \Sigma), \;\; \Sigma = \begin{pmatrix} BS^2B^T + \Delta^2 & BS\theta \\ \theta^T S B^T & \tau^2 + \theta^T\theta \end{pmatrix},$$

as is the conditional

$$Y_i \mid x_i, \theta, S, \sim \text{N}\left(\theta^T S B^T C x_i, \tau^2 + \theta^T(I - SB^TCBS)\theta\right), \quad C = (BS^2B^T + \Delta^2)^{-1} \in \mathbb{R}^p.$$

For the above model we will use two distinct simulation settings. In the first setting the dominant directions of variation in the covariates will correlate with the regression coefficient. In the second setting directions explaining less variation in the covariates will correlate with the regression coefficients – this is the least eigenvalue scenario. In both settings we set the number of factors d to be 10. The parameters $\{\theta_i\}$ and $\{s_i\}$ are drawn from a t-distribution with 5 degrees of freedom. The two settings correspond to ordering the sampled parameters such that

1. $|\theta_1| > |\theta_2| > \ldots > |\theta_d|$ and $|s_1| > |s_2| > \ldots > |s_d|$
2. $|\theta_1| < |\theta_2| < \ldots < |\theta_d|$ and $|s_1| > |s_2| > \ldots > |s_d|$.

The first setting orders the regression coefficients with the maximal eigenvalues. The second setting assigns the maximal regression coefficient to the smallest eigenvalue. The covariance structure is set to be spherical

$\Delta^2 = \psi^2 I$. The parameters ψ^2 and τ^2 control the percentage of variance explained or signal-to-noise (we set this to 0.8)

$$0.8 = \text{s2n}_x = \frac{\min(s_i^2)}{\psi^2 + \min(s_i^2)}, \quad 0.8 = \text{s2n}_y = \frac{\theta^T\theta}{\tau^2 + \theta^T\theta} = \frac{\text{Var}(\lambda_i^T\theta)}{\text{Var}(Y_i)}.$$

We used four criteria to evaluate the projections we obtained from the six methods. The first was CPU time which includes pre-processing the inputs, estimation, and post-processing of the results – denoted as time. The second was the absolute value of the correlation of the dimension reduction space which in our simulations was the vector

$$b = S_{Y|X} = \text{span}[\text{Cov}(X)^{-1}\text{Cov}(X,Y)] = \text{span}[(BS^2B^T + \Delta^2)^{-1}BS\theta].$$

We report the absolute correlation (AEDR) of b with the effective dimension reduction estimate $\hat{b}$, $|\text{corr}(b, \hat{b})|$. The third and fourth metric are based on predictive criteria. In the first case the criterion used is an estimate of the mean square prediction error (MSPE)

$$E\|Y - Z\hat{b}\|_2^2,$$

where Z is the projection of the data X onto the edr subspace and $\hat{b}$ are the regression coefficients estimates. The fourth metric is the proportion of the variance explained by the linear regression (R^2), $\text{Cor}(Y, \hat{Y})^2$, where $\hat{Y} = Z\hat{b}$. We generated 10 data sets. The performance on the different evaluation criteria was estimated using 5-fold cross-validation. For each of the two parameter settings we examined two data size regimes, $n > p$ and $p > n$. For the LSIR and SIR algorithms the number of slices is set to ten, $H = 10$, and for LSIR the nearest neighbor parameter is set to five, $k = 5$. The rank of edr subspace is set for all algorithms to $d = 1$. For the randomized SVD we use $t = 1$ power iterations and $\Delta = 5$ for the oversampling parameter. Table 2 and 3 report the results for the various methods in the setting where the signal is correlated to the top principal components and either $n > p$ or $n < p$, respectively. Table 4 and 5 report the results for the various methods in the setting where the signal is correlated to the bottom principal components and either $n > p$ or $n < p$, respectively. As expected SIR runs much faster than all other approaches since the size of the matrix decomposed is equal to the number of slices H which is much smaller than n and p. The runtime for rand.lsir tends to be comparable to SIR and is much faster than the other methods. It is known that LSIR does not perform well without adding a regularization term (Wu et al., 2010) and we reproduce this behavior. It is interesting that rand.lsir does not share this problem and does perform well without regularization, the randomization adds an implicit regularization in this case. In the setting where the regression signal is embedded into the bottom principal components randomization in LSIR has a strong effect in improving the predictive accuracy.

Method	Time	R^2	MSPE	AEDR
dr.sir	15.13 ± 0.26	0.52 ± 0.02	2.98 ± 0.10	0.08 ± 0.04
rand.sir	0.06 ± 0.00	0.78 ± 0.02	1.06 ± 0.05	0.50 ± 0.21
lsir	19.72 ± 0.08	0.22 ± 0.02	5.04 ± 0.40	0.04 ± 0.02
rand.lsir	0.08 ± 0.01	0.73 ± 0.08	1.33 ± 0.33	0.37 ± 0.16
pca	13.23 ± 0.03	0.39 ± 0.15	2.99 ± 0.68	0.37 ± 0.17
rand.pca	0.43 ± 0.00	0.39 ± 0.15	2.99 ± 0.68	0.37 ± 0.17

Table 2: Performance evaluation of dimension reduction approaches when the signal for regression is correlated with the top principal components. The simulation is based on 10 random datasets of dimension $n = 2000, p = 1000$.

Method	Time	R^2	MSPE	AEDR
rand.sir	0.06 ± 0.01	0.77 ± 0.05	1.18 ± 0.26	0.20 ± 0.08
lsir	9.38 ± 0.30	0.03 ± 0.03	4.83 ± 0.34	0.01 ± 0.01
rand.lsir	0.08 ± 0.00	0.59 ± 0.19	2.15 ± 0.96	0.08 ± 0.08
pca	13.78 ± 0.02	0.41 ± 0.14	2.99 ± 0.77	0.31 ± 0.16
rand.pca	0.41 ± 0.00	0.41 ± 0.14	$2,99 \pm 0.77$	0.31 ± 0.16

Table 3: Performance evaluation of dimension reduction approaches when the signal for regression is correlated with the top principal components. The simulation is based on 10 random datasets of dimension $n = 1000, p = 2000$.

Method	Time	R^2	MSPE	AEDR
dr.sir	15.14 ± 0.23	0.40 ± 0.03	2.93 ± 0.10	0.33 ± 0.02
rand.sir	0.06 ± 0.00	0.38 ± 0.14	2.24 ± 0.43	0.22 ± 0.09
lsir	19.69 ± 0.10	0.14 ± 0.07	3.94 ± 0.29	0.14 ± 0.05
rand.lsir	0.08 ± 0.00	0.16 ± 0.20	3.06 ± 0.83	0.36 ± 0.22
pca	13.24 ± 0.06	0.01 ± 0.01	3.59 ± 0.27	0.00 ± 0.01
rand.pca	0.43 ± 0.01	0.01 ± 0.01	3.59 ± 0.27	0.00 ± 0.01

Table 4: Performance evaluation of dimension reduction approaches when the signal for regression is correlated with the bottom principal components. The simulation is based on 10 random datasets of dimension $n = 1000, p = 2000$.

Method	Time	R^2	MSPE	AEDR
rand.sir	0.06 ± 0.00	0.09 ± 0.04	3.44 ± 0.47	0.09 ± 0.04
lsir	9.46 ± 0.19	0.04 ± 0.03	3.53 ± 0.39	0.05 ± 0.02
rand.lsir	0.08 ± 0.01	0.16 ± 0.03	3.21 ± 0.45	0.15 ± 0.05
pca	13.80 ± 0.02	0.01 ± 0.00	3.65 ± 0.46	0.01 ± 0.01
rand.pca	0.42 ± 0.01	0.01 ± 0.00	3.65 ± 0.46	0.01 ± 0.01

Table 5: Performance evaluation of dimension reduction approaches when the signal for regression is correlated with the bottom principal components. The simulation is based on 10 random datasets of dimension $n = 2000, p = 1000$.

3.1.3 Classification example

This section demonstrates the performance of the dimension reduction approaches on the XOR classification example from (Wu et al., 2010), Section 4, (Fig 1). The dimension reduction subspace is two-dimensional, symmetric, with clustering structure. SIR is expected to fail to recover the edr subspace because $E[X|Y] \subset S_{Y|X}$ – the edr subspace inferred by SIR in this case is degenerate. Figure 1 and

2 illustrate the results when $n > p$ and $p > n$, respectively. As expected, SIR fails to recover a two-dimensional subspace, while both LSIR and rand.lsir provide reasonable separation when $n > p$. Only LSIR with randomization performs well in the more difficult setting when $p > n$ – in (Wu et al., 2010), Secton 4, LSIR with regularization performed well in this same $p > n$ however rand.lsir performs well even without adding a regularization term.

Figure 1: XOR classification example. n=3000, p=1000

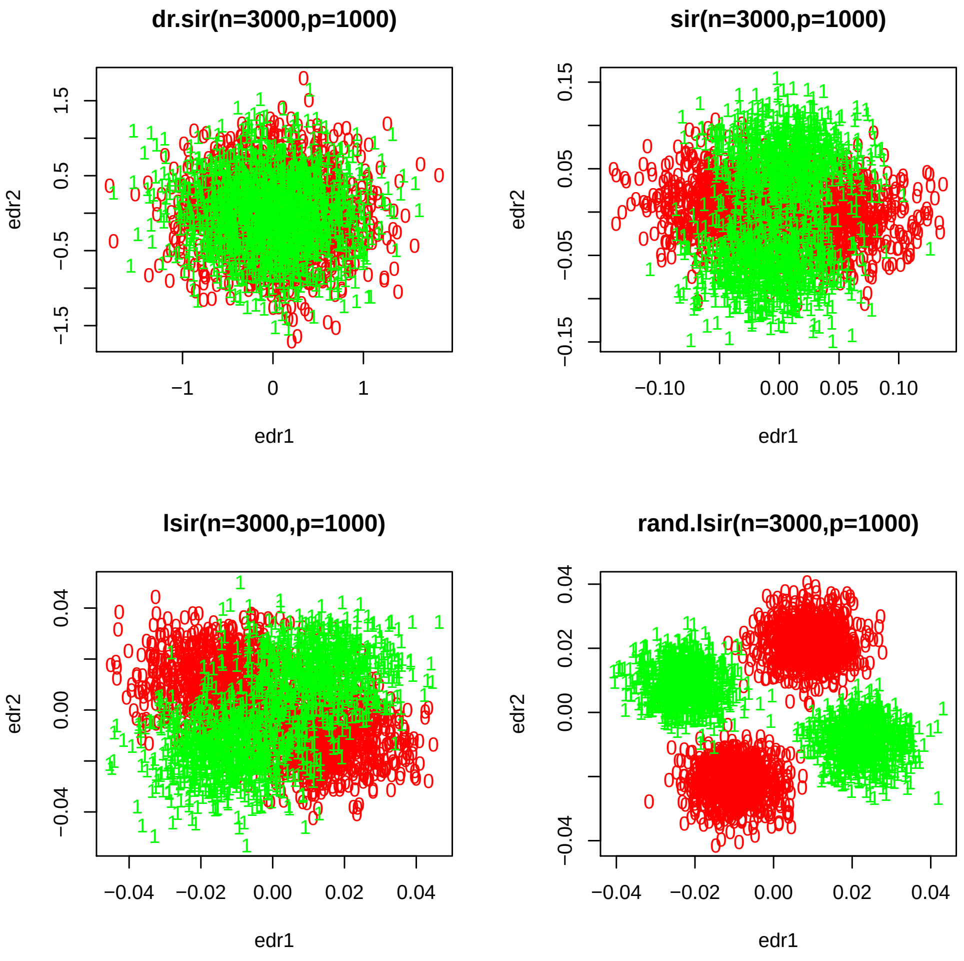
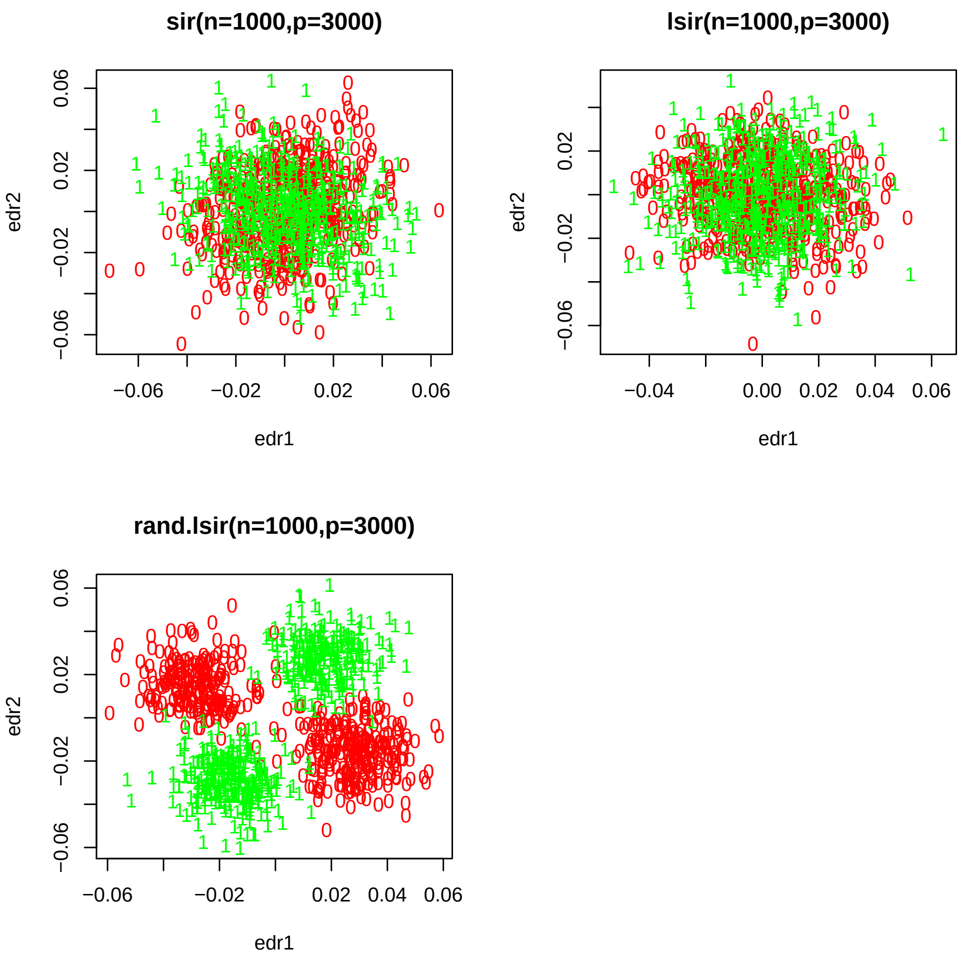


Figure 2: XOR classification example. n=1000, p=3000



3.2 Real data

3.2.1 Digit Recognition

The MNIST data set (Y. LeCun, htpp://yann.lecun.com/exdb/mnist/) contains 60,000 images of handwritten grey-scale digits of dimension $p = 28 \times 28 = 728$. This data set has been extensively studied in the past and has been found to contain non-linear structure. The dimension reduction methods we compare are PCA, rand.pca, SIR, LSIR, rand.lsir, and random projections (rand.proj) on this data set. We use classification accuracy as our metric and use a 5-nearest neighbor classifier in the edr space as the classification algorithm. We set the dimension of the edr for all the methods to $d = 20$. For the randomized methods the oversampling parameter $\Delta = 5$ and the number of iterations $t = 1$. For LSIR we set the number of nearest neighbors $k = 5$. The training and test sets consist of 100 randomly selected images of each digit, respectively. The performance results are summarized in Table 6. As in the previous section we see that the randomized supervised methods outperform their exact analogs. We suspect this is again due to their implicit regularization. Note that LSIR properly tuned will do better than SIR on this classification task (Wu et al., 2010) – in our implementation of LSIR we did not include the regularization parameter.

Digits	0	1	2	3	4	5	6	7	8	9
pca	96.0 ± 1.6	98.9 ± 0.7	88.6 ± 3.5	86.3 ± 4.4	85.0 ± 5.2	85.3 ± 1.9	95.1 ± 2.4	90.0 ± 2.5	80.4 ± 4.0	83.2 ± 4.5
rand.pca	96.0 ± 1.8	98.8 ± 0.8	88.6 ± 3.6	86.9 ± 4.0	83.9 ± 5.5	86.1 ± 3.1	95.5 ± 2.5	89.5 ± 2.7	80.5 ± 3.7	84.0 ± 3.6
dr.sir	91.2 ± 1.9	97.7 ± 1.4	51.3 ± 7.6	67.6 ± 7.5	70.8 ± 5.7	56.6 ± 7.5	81.5 ± 2.8	71.8 ± 5.9	55.0 ± 4.9	72.2 ± 6.2
sir	90.0 ± 3.4	96.6 ± 2.0	73.3 ± 4.9	76.2 ± 3.8	82.9 ± 2.9	67.7 ± 3.2	87.0 ± 3.8	80.4 ± 4.1	69.2 ± 5.0	74.4 ± 5.4
lsir	75.5 ± 5.0	95.7 ± 1.6	38.2 ± 7.5	52.4 ± 8.1	52.2 ± 4.4	39.6 ± 6.2	$61.0 \pm 5.4.8$	58.3 ± 6.2	36.3 ± 4.1	56.2 ± 6.5
rand.lsir	92.2 ± 3.0	98.7 ± 1.3	81.9 ± 5.7	82.1 ± 4.2	78.9 ± 2.7	76.4 ± 6.9	89.1 ± 3.9	82.2 ± 2.9	66.9 ± 3.8	78.8 ± 5.8
rand.proj	86.1 ± 3.8	98.6 ± 1.3	54.1 ± 10.1	66.2 ± 8.6	65.9 ± 7.6	47.9 ± 7.6	77.4 ± 6.9	77.7 ± 5.3	43.4 ± 5.7	61.9 ± 7.0

Table 6: Digit recognition (% accuracy). 100 train and 100 test examples, randomly sampled for each digit.

3.2.2 Tumor classification

The gene expression data on 198 tumor samples from Ramaswamy et al. (2001), including 14 common tumor types, was used in conjunction with 5NN classifier to evaluate the different dimension reduction approaches. The full set of 16,063 gene and expressed sequence tags, without any additional pre-processing, was used for the purposes of the comparison. The predictive accuracy was estimated as the average over 10 random binary train/test partitions of the data, requiring at least 5 samples from each class to be present in both the train and the test set. For the randomized methods we set the oversampling parameter $\Delta = 5$, the number of power iterations $t = 1$, and the number of nearest neighbors $k = 5$. The number of edr directions we used were $d = 12$. Table 7 states the results for the various dimension reduction methods. Note that SIR and rand.lsir outperformed the other approaches.

Method	pca	rand.pca	sir	lsir	rand.lsir	rand.proj
mean	45.15 ± 4.97	45.15 ± 5.45	57.78 ± 3.95	11.62 ± 2.20	54.14 ± 4.45	39.60 ± 4.94

Table 7: Gene expression data from the study of 14 common tumor types Ramaswamy et al. (2001). The input data set was randomly Results were generated based on 10 random splits into train and test set and reporting the average.

4 Discussion

We adapted randomized SVD to improve the runtime of a variety of dimension reduction methods. The main algorithmic contribution of our work is to develop very fast randomized algorithms for a class of generalized eigendecompositions which arise in a variety of dimension reduction methods. From an inference perspective we observe that randomization imposes implicit regularization on the resulting estimators. This is highlighted in the supervised dimension reduction case corresponding to the least eigenvalue situation – the regression signal is contained in the eigenvectors corresponding to small eigenvalues. The exact role of the number of iterations t as a regularization parameter is an open question. As t gets large the subspace of the approximated algorithm converges very quickly to the subspace of the exact algorithm. Hence, in order to apply the proposed algorithms to massive data sets we need t to be small. A statistical argument for few iterations can also be made based on a similar idea to ridge shrinkage. The main effect of the randomized algorithm is a reweighting of the eigenvalues $F^{(t)} = US^{2t}V^T\Omega$, where the reweighting comes from S^{2t}. If t is small then the eigenvalues are shrunk much less, this is similar in spirit to flattening the eigenvalues. This allows for more mixing between the eigenvectors and much as ridge regression it adds “signal” to the lower eigenvalues. This is related to the classic least eigenvalue problem (Hotelling, 1957; Cox, 1968) where the main regression signal is in the principal components corresponding to the smaller eigenvalues. Both the randomized algorithm with small t and ridge regularization will help address the least eigenvalue problem by boosting the weights of the directions corresponding to smaller eigenvalues. It would be of great interest to develop some theoretical analysis of this setting.

Acknowledgements

SM would like to acknowledge Lek-Heng Lim, Michael Mahoney, Qiang Wu, and Ankan Saha. SM is pleased to acknowledge support from grants NIH (Systems Biology): 5P50-GM081883, AFOSR: FA9550-10-1-0436, NSF CCF-1049290, and NSF-DMS-1209155. SG would like to acknowledge Uwe Ohler and Ankan Saha.

References

Achlioptas, D. (2001). Database-friendly random projections. In *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, PODS ’01, New York, NY, USA, pp. 274–281. ACM.

Adcock, R. (1878). A problem in least squares. *The Analyst 5*, 53–54.

Anderson, E., Z. Bai, J. Dongarra, A. Greenbaum, A. McKenney, J. Du Croz, S. Hammerling, J. Demmel, C. Bischof, and D. Sorensen (1990). Lapack: a portable linear algebra library for high-performance computers. In *Proceedings of the 1990 ACM/IEEE conference on Supercomputing*, Supercomputing ’90, Los Alamitos, CA, USA, pp. 2–11. IEEE Computer Society Press.

Belkin, M. and P. Niyogi (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation 15*(6), 1373–1396.

Belkin, M. and P. Niyogi (2005). Towards a theoretical foundation for laplacian-based manifold methods.

Boutsidis, C., M. W. Mahoney, and P. Drineas (2009). An improved approximation algorithm for the column subset selection problem. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA ’09, pp. 968–977. Society for Industrial and Applied Mathematics.

Chung, F. R. K. (1997). *Spectral Graph Theory*, Volume 92. American Mathematical Society.

Cook, R. (2007). Fisher lecture: Dimension reduction in regression. *Statistical Science 22*(1), 1–26.

Cook, R. and S. Weisberg (1991). Disussion of li (1991). *J. Amer. Statist. Assoc. 86*, 328–332.

Cox, D. (1968). Notes on some aspects of regression analysis. *Journal of the Royal Statistical Society Series A 131*, 265–279.

Dasgupta, S. and A. Gupta (2003). An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms 22*(1), 60–65.

Donoho, D. and C. Grimes (2003). Hessian eigenmaps: new locally linear embedding techniques for highdimensional data. *PNAS 100*, 5591–5596.

Drineas, P., R. Kannan, and M. W. Mahoney (2006, July). Fast monte carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix. *SIAM J. Comput. 36*, 158–183.

Edegworth, F. (1884). On the reduction of observations. *Philosophical Magazine*, 135–141.

Fisher, R. (1922). On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Statistical Society A 222*, 309–368.

Fisher, R. (1936). The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics 7*(2), 179–188.

Frankl, P. and H. Maehara (1987, June). The johnson-lindenstrauss lemma and the sphericity of some graphs. *J. Comb. Theory Ser. A 44*, 355–362.

Globerson, A. and S. Roweis (2006). Metric learning by collapsing classes. In Y. Weiss, B. Schölkopf, and J. Platt (Eds.), *Advances in Neural Information Processing Systems 18*, pp. 451–458. Cambridge, MA: MIT Press.

Goldberger, J., S. Roweis, G. Hinton, and R. Salakhutdinov (2005). Neighbourhood component analysis. In *Advances in Neural Information Processing Systems 17*, pp. 513–520.

Golub, G., K. Slna, and P. V. Dooren (2000). Computing the svd of a general matrix product/quotient. *SIAM J. Matrix Anal. Appl 22*, 1–19.

Golub, G. H. (1969). Matrix decompositions and statistical calculations. Technical report, Stanford, CA, USA.

Golub, G. H. and C. F. Van Loan (1996). *Matrix computations* (3rd ed.). Baltimore, MD: John Hopkins University Press.

Gu, M. and S. C. Eisenstat (1996, July). Efficient algorithms for computing a strong rank-revealing qr factorization. *SIAM J. Sci. Comput. 17*(4), 848–869.

Halko, N., P. Martinsson, and J. A. Tropp (2009, September). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *ArXiv e-prints*.

Hastie, T. and R. Tibshirani (1996). Discrminant adaptive nearest neighbor classification. *IEEE Transacations on Pattern Analysis and Machine Intelligence 18*(6), 607–616.

He, X. and P. Niyogi (2003). Locality preserving projections. In *NIPS*.

Hotelling, H. (1933). Analysis of a complex of statistical variables in principal components. *Journal of Educational Psychology 24*, 417–441.

Hotelling, H. (1957). The relationship of the newer multivariate statistical methods to factor analysis. *British Journal of Statistical Psychology 10*, 69–79.

Indyk, P. and R. Motwani (1998). Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, STOC '98, New York, NY, USA, pp. 604–613. ACM.

Johnson, W. and J. Lindenstrauss (1984). Extensions of Lipschitz mappings into a Hilbert space. In *Conference in modern analysis and probability (New Haven, Conn., 1982)*, Volume 26 of *Contemporary Mathematics*, pp. 189–206. American Mathematical Society.

Jolliffe, I. T. (1982). A note on the use of principal components in regression. *Journal of the Royal Statistical Society, Series C 31*(3), 300–303.

Li, B., H. Zha, and F. Chiaromonte (2005). Contour regression: A general approach to dimension reduction. *The Annals of Statistics 33*(4), 1580–1616.

Li, K. (1991). Sliced inverse regression for dimension reduction (with discussion). *J. Amer. Statist. Assoc. 86*, 316–342.

Li, K. C. (1992). On principal hessian directions for data visulization and dimension reduction: another application of stein's lemma. *J. Amer. Statist. Assoc. 87*, 1025–1039.

Liberty, E., F. Woolfe, P.-G. Martinsson, V. Rokhlin, and M. Tygert (2007). Randomized algorithms for the low-rank approximation of matrices. *Proceedings of the National Academy of Sciences 104*(51), 20167–20172.

Mahoney, M. W. (2011). Randomized algorithms for matrices and data. *CoRR abs/1104.5557*.

Martinsson, P., A. Szlam, and M. Tygert (Vancouver, Canada, 2010). Normalized power iterations for the computation of SVD. *NIPS workshop on low-rank methods for large-scale machine learning*.

Nilsson, J., F. Sha, and M. Jordan (2007). Regression on manifolds using kernel dimension reduction. In *Proceedings of the 24th International Conference on Machine Learning*.

Polson, N. G. and J. G. Scott (2010). Shrink globally, act locally: Sparse bayesian regularization and prediction. In *Bayesian Statistics 9*. Oxford University Press.

Ramaswamy, S., P. Tamayo, R. Rifkin, S. Mukherjee, C.-H. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. P. Mesirov, T. Poggio, W. Gerald, M. Loda, E. S. Lander, and T. R. Golub (2001). Multiclass cancer diagnosis using tumor gene expression signatures. *Proceedings of the National Academy of Sciences 98*(26), 15149–15154.

Rokhlin, V., A. Szlam, and M. Tygert (2008, September). A randomized algorithm for principal component analysis. *ArXiv e-prints*.

Roweis, S. and L. Saul (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science 290*, 2323–2326.

Sarlos, T. (2006, oct.). Improved approximation algorithms for large matrices via random projections. In *Foundations of Computer Science, 2006. FOCS '06. 47th Annual IEEE Symposium on*, pp. 143 –152.

Sugiyama, M. (2007). Dimension reduction of multimodal labeled data by local fisher discriminatn analysis. *Journal of Machine Learning Research 8*, 1027–1061.

Tenenbaum, J., V. de Silva, and J. Langford (2000). A global geometric framework for nonlinear dimensionality reduction. *Science 290*, 2319–2323.

Weisberg, S. (2002). Dimension reduction regression in r. *J. Statistical Software 7*.

West, M. (2003). Bayesian Factor Regression Models in the "Large p, Small n" Paradigm. *Bayesian Statistics 7*, 723–732.

Wu, Q., F. Liang, and S. Mukherjee (2010). Localized sliced inverse regression. *Journal of Computational and Graphical Statistics 19*(4), 843–860.

Young, G. (1941). Maximum likelihood estimation and factor analysis. *Psychometrika 6*, 49–53.